MODERN APPLICATION DEVELOPMENT - PROJECT

Quiz Master Platform Report

Name: Kandula Sandeep Kumar
Email: 23f3003779@ds.study.iitm.ac.in
Roll Number: 23f3003779

I am Kandula Sandeep Kumar, currently pursuing my diploma degree from **IITM** and majoring in Mechanical Engineering at **NIT Warangal**. I am excited to submit my **MAD-1** project on **Quiz Master.**

1) Description

Quiz Master is a comprehensive web-based assessment platform designed for educational environments. It provides a structured system for creating, managing, and taking quizzes across various subjects and topics. The application features role-based access with separate interfaces for administrators (quiz creators) and students (quiz takers), allowing for efficient knowledge assessment and progress tracking.

2) Technologies Used

- **Backend Framework**: Flask (Python)
- **Database and ORM**: Sqlite, SQLAlchemy
- **Frontend**: HTML5, CSS3, Bootstrap 5, JavaScript
- **Authentication, Form Handling**:: Flask-Login**,** Flask-WTF
- **Password Security**: Werkzeug Security
- **Visualization and Icons:**: Chart.js, Font Awesome

3) DB Schema Design

The database is organized in a hierarchical structure with the following key models:

- **User**: Stores user account data with authentication details and role information
  - Fields: id, username, password_hash, full_name, qualification, dob, is_admin
- **Subject**: Top-level content organization
  - Fields: id, name, description
- **Chapter**: Second-level content organization within subjectsFields: id, subject_id, name, description
- **Quiz**: Assessment units within chapters
  - Fields: id, chapter_id, title, description, date, duration
- **Question**: Individual questions within quizzes
  - Fields: id, quiz_id, question_text, options, correct_answer
- **Score**: Records of quiz attempts by users
  - Fields: id, quiz_id, user_id, timestamp, total_score, correct_answers, total_questions
- **UserAnswer**: Individual responses to questions
  - Fields: id, score_id, question_id, user_answer

Entity-Relationship (ER) Diagram

```
+----------------+          +----------------+          +----------------+
|      User      |          |    Subject     |          |    Chapter     |
+----------------+          +----------------+          +----------------+
| id (PK)        |          | id (PK)        |          | id (PK)        |
| username       |          | name           |          | subject_id (FK)|
| password_hash  |          | description    |          | name           |
| is_admin       |          |                |          | description    |
+----------------+          +----------------+          +----------------+
       |                           |                           |
       v                           v                           v
+----------------+          +----------------+          +----------------+
|     Score      |          |      Quiz      |          |    Question    |
+----------------+          +----------------+          +----------------+
| id (PK)        |          | id (PK)        |          | id (PK)        |
| quiz_id (FK)   |<-----> | chapter_id (FK)|   <----->  | quiz_id (FK)   |
| user_id (FK)   |          | title          |          | question_text  |
| total_score    |          | description    |          | options        |
| correct_answers|          | duration       |          | correct_answer |
+----------------+          +------------------+        +----------------+
        |
        V
+----------------+
|  UserAnswer    |
+-----------------------+
| id (PK)               |
| score_id (FK)         |
| question_id (FK)      |
| user_answer           |
+-----------------------+
```

4) Architecture and Features

Quiz Master follows the Model-View-Controller (MVC) architectural pattern:

- **Models**: SQLAlchemy models define the database structure and relationships
- **Views**: Jinja2 templates render the user interface
- **Controllers**: Flask routes handle request processing and business logic

5) Key Features

- **Role-based Access Control**: Separate interfaces for admins and students
- **Hierarchical Content Organization**: Subjects → Chapters → Quizzes → Questions
- **Secure Authentication**: Password hashing and form protection
- **Responsive Design**: Mobile-friendly interface
- **Real-time Quiz Taking**: Interactive quiz interface with timer
- **Performance Analytics**: Detailed statistics and visualizations

- **Search Functionality**: Content and user search capabilities
- **CRUD Operations**: Complete management interfaces for all content types

6) Admin Functionality

- **Dashboard**: Overview with key statistics and platform activity
- **Subject Management**: Create, view, edit, and delete subject areas
- **Chapter Management**: Organize content within subjects
- **Quiz Creation**: Set up timed assessments with custom parameters
- **User Management**: View user profiles, track performance, and manage accounts
- **Analytics**: Visual representations of platform usage and performance metrics
- **Search and Filter**: Find specific content or users quickly

7) User Functionality

- **Registration**: Self-service account creation
- **Dashboard**: Personalized overview with performance metrics
- **Quiz Catalog**: Browse available quizzes with filtering options
- **Quiz Taking**: Interactive interface with timer and progress tracking
- **Results Review**: Detailed feedback after quiz completion
- **Performance History**: Track scores and progress over time

8) API Endpoints

8.1 Authentication

- GET/POST /login - User login.
- GET/POST /register - User registration.
- GET /logout - User logout.

8.2 Admin

- GET /admin/dashboard - Dashboard with statistics.
- GET/POST /admin/subjects - Manage subjects.
- GET/POST /admin/quizzes - Manage quizzes.
- GET /admin/analytics - View analytics.

8.3 Student

- GET /user/dashboard - View quizzes.
- GET /user/take_quiz/<quiz_id> - Take a quiz.
- POST /user/submit_quiz/<quiz_id> - Submit answers.
- GET /user/quiz_results/<score_id> - View results

Video Presentation Link: Link